Implementasi Algoritme F5 untuk Penyisipan Pesan Rahasia pada Citra Digital

Maria Magdalena[1], Nikolaus Adi Putra[2], Eka Puji Widiyanto[3], Willy[4]
[1,2,3,4] Jurusan Informatika, STMIK GI MDP Palembang
Jl. Rajawali No.14 Palembang
Telp. (0711) 376400
Email : [1]maria.mm07@mhs.mdp.ac.id, [2]nicko_ap0000@mhs.mdp.ac.id,
[3]ekapujiw2002@gmail.com, [4]dominikuswilly@yahoo.com

Abstrak

Steganografi dapat menjadi solusi untuk pengamanan data saat melakukan pertukaran data secara online. Teknik steganografi dapat diterapkan menggunakan algoritme F5 untuk meningkatkan keamanan pesan yang terkandung dalam media citra digital format JPEG. Hal ini dikarenakan algoritme F5 menggunakan permutasi dengan menyebar bit-bit pesan kesuluruh citra digital sehingga tidak menimbulkan kecurigaan akan keberadaan pesan di dalam media tersebut. Maka dari itu, penelitian ini dilakukan untuk mengamankan pesan rahasia pada citra digital dengan format JPEG menggunakan algoritme F5. Berdasarkan hasil pengujian encode pesan, didapatkan hubungan antara waktu dan besarnya resolusi citra yang digunakan. Semakin besar resolusi citra, maka semakin banyak waktu yang dibutuhkan untuk melakukan encode pesan. Berdasarkan hasil pengujian error, stego-image memiliki nilai Mean Square Error (MSE) yang rendah yaitu berkisar antara 5 - 13 dan nilai Peak Signal to Noise Ratio (PSNR) berkisar antara 39 – 41 dB yang berarti stego-image memiliki kualitas citra yang baik karena nilai kuadrat error yang rendah. Kelemahan algoritme F5 yaitu tidak robust terhadap penambahan image distortion seperti Gaussian Blur, rotasi citra (rotation), pemotongan citra (cropping), dan perubahan ukuran (scaling).

Kata kunci — Steganografi, algoritme F5, pesan rahasia, citra digital, JPEG

Abstract

Steganography can be a good solution for securing data when do the exchange data online. Steganography technique can be applied using F5 algorithm to improve the security of the messages that contained in the digital image media with JPEG format. It cause the F5 algorithm using the permutation to spread the message bits over the whole of digital image, so we can't detect that message. Therefore, the goal of this paper is to protect message on the digital image media with JPEG format without evoked the suspicion. Based on the result of message encode testing, there is a correlation between time and size of image resolution. The greater resolution of the image, the more time it takes to encode the message. Based on the result of error testing, the stego-image has a low MSE value between 5 - 13 and PSNR value between 39 – 41 dB, it means that stego-image has good image quality because the value of MSE is low. The weakness of F5 algorithm is not robust to the addition of image distortion such as Gaussian Blur, image rotation, cropping, and scaling.

Keywords— Steganography, F5 algorithm, secret message, digital image, JPEG

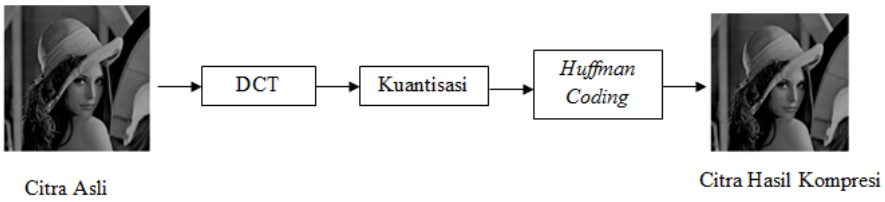
1. PENDAHULUAN

Pertukaran informasi secara *online* yang bersifat rahasia masih belum dikatakan aman. Hal ini dikarenakan pihak yang tidak memiliki hak atas informasi yang terkandung di dalamnya dapat dengan mudah mengakses informasi tersebut sehingga timbul masalah keamanan yang kurang dalam pertukaran informasi. Maka dari itu diperlukan suatu teknik pengiriman data yang aman sehingga tidak menimbulkan kecurigaan.

Teknik yang bisa digunakan untuk pengamanan data diantaranya kriptografi dan steganografi. Kriptografi merupakan teknik pengenkripsian, namun kurang tepat digunakan karena teknik pengacakan pesan dapat menimbulkan kecurigaan. Dalam contoh kasus di atas, steganografi cukup aman untuk diterapkan dikarenakan tujuan steganografi yaitu untuk mengamankan data dengan menyembunyikan isi pesan dalam suatu media, sehingga hanya pihak terkait saja yang mengetahui adanya pesan rahasia dengan syarat pengirim dan penerima memasukkan *password* yang sama.

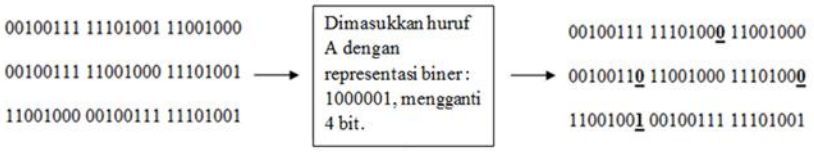
Kata steganografi terdiri dari 2 (dua) penggalan kata yaitu *steganos* dan *graphein* yang berarti "tulisan tersembunyi" [1]. Steganografi sendiri merupakan suatu ilmu yang mempelajari bagaimana cara menyembunyikan suatu pesan rahasia pada suatu media sehingga hanya pihakyang terkait saja yang mengetahui isi pesan rahasia tersebut. Steganografi menyediakan format media digital untuk penyisipan pesan yang beragam seperti format *image* (jpeg, bitmap, dan tiff), format *audio* (wav, voc, dan mp3), dan format lain seperti teks file,html, pdf, dll [2].

Untuk menghindari kecurigaan, steganografi dapat diterapkan pada media yang umum digunakan pada pertukaran data digital, yaitu media citra digital. Format citra digital yang banyak digunakan yaitu citra dengan format JPEG. Citra JPEG yang telah disisipi pesan rahasia tidak terlihat berbeda secara kasat mata, maka dari itu mengurangi tingkat kecurigaan pihak yang tidak bertanggungjawab. Format JPEG saat ini format yang paling umum untuk menyimpan data gambar. Kompresi dengan JPEG menggunakan beberapa proses, yaitu DCT (*Discrete Cosine Transform*), kuantisasi, dan penyandian entropi (*Huffman Coding*) [3].



Gambar 1 Proses Standar Kompresi JPEG

Ada banyak metode steganografi pada citra digital yang sudah berkembang, diantaranya adalah menggunakan metode LSB [2,5], *Discreate Wavelet Transform* [6], 2-Level *Discrete Wavelet Transform* (DWT) [7], dan metode F5 [7, 8, 9, 10 11, 12]. Metode yang paling umum dan sering digunakan yaitu metode LSB [13]. Metode LSB cukup sederhana, karena LSB menyisipi pesan hanya dengan cara mengganti bit yang letaknya di sebelah kanan dari barisan bit pada representasi biner gambar dengan representasi biner pesan rahasia yang akan disembunyikan [2]. Untuk mengetahui cara kerja dari LSB, dapat dilihat pada Gambar 2.



Gambar 2 Contoh Penggantian Bit dengan LSB [13]

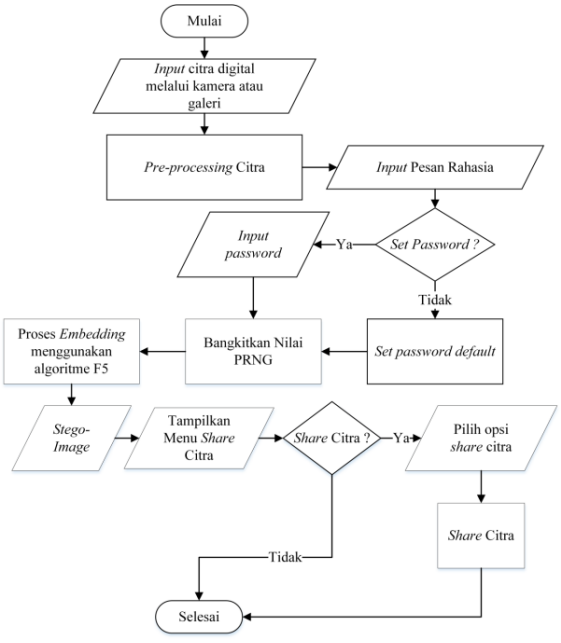
Penelitian ini menggunakan algoritme F5 yang merupakan perbaikan dari algoritme F3 dan F4 oleh peneliti yang sama yaitu Andreas Westfeld. Kelebihan dari algoritme ini yaitu penyebaran pesannya lebih merata keseluruh media citra penampung (*cover-image*) karena menggunakan permutasi sehingga keberadaan pesan sulit untuk terdeteksi [9], selain itu F5 menawarkan kapasitas penyimpanan data besar dengan proporsi pesan yang ditampung sebesar 13% dari citra penampungnya [10]. Algoritme F5 dapat mencegah serangan statistikdan meningkatkan efisiensi penyisipan karena memiliki 2 fitur utama yaitu *Permutative Straddling* dan *Matrix Encoding* [11].

Penelitian ini mengacu kepada penelitian yang pernah dilakukan sebelumnya oleh Octavianus [12] menggunakan algoritme yang sama. Perbedaan dengan penelitian sebelumnya yaitu pengujian dilakukan lebih dalam untuk mengetahui hubungan waktu *encode* dengan resolusi citra dan banyaknya karakter yang diinput, serta menguji ketahanan dari citra yang telah disisipi pesan (*stego-image*) jika dilakukan operasi manipulasi citra seperti menambahkan efek *Gaussian Blur*, *cropping*, rotasi, dan *scalling*. Selain itu, penelitian ini menguji ulang ketahanan citra yang dilakukan oleh Zulfikar [10] untuk mengetahui bahwa algoritme F5 tidak *robust* terhadap manipulasi operasi citra.

Pada penelitian ini digunakan citra berwarna RGB dengan format JPEG, data yang disisipkan berupa teks dengan panjang maksimal 1000 karakter (“a - z”, “A - Z”, “0 – 9”). Selain itu, format citra yang digunakan sebagai media penyisipan berukuran minimal 640 x 480 piksel dan maksimal 3264 x 2448 piksel. Penelitian ini menggunakan *platform* Android versi 4.2.2 *Jelly Bean* untuk membangun aplikasi.

2. METODE PENELITIAN

Metode penelitian yang dilakukan untuk melakukan implementasi algoritme F5 untuk penyisiapn pesan rahasia pada citra digital sesuai dengan gambar 3. Proses pertama diawali dengan proses *input* citra digital melalui galeri kamera atau galeri. Selanjutnya citra yang telah di-*input* akan diproses oleh aplikasi.



Gambar 3 *Flowchart* Sistem

Tahapan selanjutnya yaitu *pre-processing* citra sesuai dengan gambar 4 untuk mempersiapkan citra yang digunakan agar dapat diproses pada tahap *embedding*.

Gambar 4 Tahapan *Pre-processing* Citra

Nilai RGB dibaca setelah pengguna memilih citra, kemudian nilai tersebut akan dikonversi menjadi YCbCr yang bertujuan untuk membuang komponen yang tidak penting pada citra sesuai dengan tingkat kepekaan mata manusia, sehingga menghemat ruang pada citra. Mata manusia lebih peka terhadap perubahan warna *luminance* (Y) daripada warna *chrominance* (Cb, Cr) sehingga yang digunakan untuk masukkan citra adalah warna Y [10]. Konversi warna RGB menjadi YCbCr menggunakan persamaan [5] (1), (2), (3) :

$$Y = 0.299\,R + 0.587\,G + 0.114\,B \quad (1)$$

$$Cb = -0.1687\,R - 0.3313\,G + 0.5\,B + 128 \quad (2)$$

$$Cr = 0.5\,R - 0.418\,G - 0.0813\,B + 128 \quad (3)$$

Tahap selanjutnya yaitu membagi citra ke dalam blok 8×8 yang berfungsi untuk mempermudah proses DCT. Selanjutnya untuk mendapat 64 koefisien DCT digunakan persamaan [10] (4):

$$F(u,v) = \frac{2}{8}\,C(u)C(v)\left[\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y)\cos\frac{(2x+1)u\pi}{16}\cos\frac{(2y+1)v\pi}{16}\right] \quad (4)$$

Pada persamaan di atas, $F(u,v)$ berbentuk matriks 2 dimensi 8×8, dimana :

a. $u, x = 0, 1, 2, \ldots, n-1; v, y = 0, 1, 2, \ldots, m-1$;
b. u, v merupakan koordinat frekuensi pada domain transformasi atau koefisien-koefisien DCT;
c. x, y adalah koordinat spasial dari domain asal
d. $C(u), C(v) = \frac{1}{\sqrt{2}}$ untuk u = 0;

Tahapan ini untuk mengidentifikasi dan menghilangkan komponen pada citra berfrekuensi tinggi yang tidak dapat dideteksi oleh mata manusia namun tidak mengurangi kualitas citra. 64 koefisien DCT ini yang akan digunakan untuk menyisipkan bit-bit pesan rahasia yang dimasukkan oleh pengguna. Setelah proses DCT dilakukan, 64 koefisien DCT akan dibagi dengan koefisien tabel kuantisasi *luminance* dan *chrominance* sesuai pada tabel 1.

Tabel 1 Kuantisasi *Luminance* dan *Chrominance* [9]

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	90

17	18	24	47	99	99	99	99
18	21	26	66	99	99	99	99
24	26	56	99	99	99	99	99
47	66	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99

Proses pembagian 64 koefisien DCT dengan tabel 1 menggunakan rumus persamaan [10] (5):

$$F_Q(u,v) = Round\left[\frac{F[u,v]}{Q[u,v]}\right] \quad (5)$$

Koefisien tabel kuantisasi dinyatakan dengan $Q[u, v]$, dan $F[u, v]$ merupakan 64 koefisien DCT. Tujuan dari tahapan ini yaitu untuk membuang koefisien-koefisien hasil DCT yang tidak diperlukan dalam pembentukan citra baru.

Tahapan kompresi JPEG berhenti sementara pada tahap kuantisasi. Selanjutnya dilakukan proses *embedding* pesan sesuai dengan gambar 5. Pada tahap awal, akan dibangkitkan *random number* yang *seed*-nya diperoleh dari masukkan *password* pengguna atau akan menggunakan *default seed* jika pengguna tidak memasukkan *password*. Setelah *random number* didapat, maka dilakukan permutasi menggunakan *random number* dan koefisien DCT yang telah diperoleh sebelumnya. Setelah itu, ditentukan nilai k dari kapasitas *embedding* pada data citra dan dari panjang data pesan yang akan disisipkan. Nilai k yang diperoleh akan digunakan untuk menentukan panjang dari *code word* (*array* yang menampung koefisien *non zero*) dengan rumus $n = 2^k - 1$.

Gambar 5 Tahapan *Embedding* Pesan

Tahapan selanjutnya yaitu menyisipkan pesan rahasia dengan $(1, n, k)$ *matrix encoding* dengan ketentuan: (i) isi *buffer* dengan koefisien n *non zero*, (ii) lakukan proses *hashing* pada *buffer* (menghasilkan nilai *hash* dengan k *bit-places*), (iii) tambahkan bit k berikutnya dari data pesan pada nilai *hash* (lakukan bit per bit dengan operator XOR), (iv) jika hasil yang di dapat sama dengan 0, maka nilai *buffer* dibiarkan tetap dan tidak diubah. Tetapi, jika hasil yang didapat sama dengan nilai rentang *index buffer* yaitu $1 \ldots n$, maka nilai absolut dari elemen pada *index* tersebut harus dikurangi 1, (v) lakukan pengecekan jika koefisien yang diubah tidak sama dengan 0. Jika sama, maka terjadi proses *shrinkage*. Jika peristiwa ini terjadi, maka tambahkan satu koefisien *non zero* pada *buffer* dan hilangkan koefisien 0 tadi. Lalu ulangi langkah(i), (vi) jika tidak terjadi peristiwa *shrinkage*, maka isi *buffer* dengan koefisien DCT selanjutnya (dimulai dari *index* koefisien ditambah 1). Jika masih ada data pesan yang akan disisipkan, maka ulangi langkah (i).

Jika semua tahapan telah dilakukan, maka dilanjutkan dengan proses *Huffman Coding.* Tujuan dilakukannya kompresi ini yaitu untuk melakukan kompresi citra agar ukuran yang dihasilkan tidak terlalu besar. Hasil akhir dari tahapan ini yaitu *stego-image* yang telah terkompresi.

Gambar 6 Tahapan *Decode* Pesan

Proses dari *decode* pesan adalah kebalikan dari proses *encode*. Syarat melakukan *decode* pesan yaitu password yang dimasukkan saat *encode* pesan harus sama saat *decode* pesan. Urutan dari proses *decode* dapat dilihat pada gambar 5. Langkah pertama setelah pengguna memasukkan *password* yang benar yaitu melakukan dekompres pada *stego-image* menggunakan *Huffman Decoder*, selanjutnya dilakukan permutasi terhadap koefisien DCT dan *random number* yang dibangkitkan pada saat memasukkan *password*. Selanjutnya pesan akan di-*decode* dan menghasilkan uraian pesan.

3. HASIL DAN PEMBAHASAN

3.1 Pengujian Encode Pesan

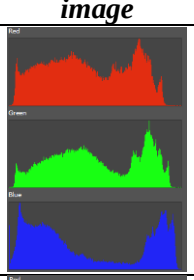
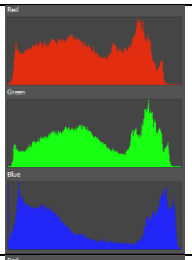
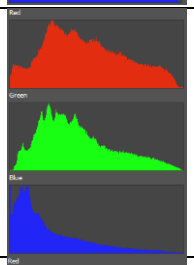
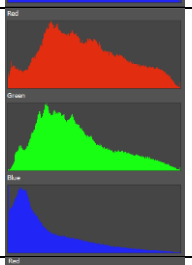
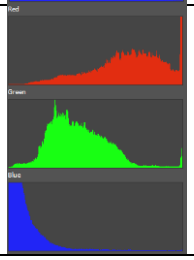
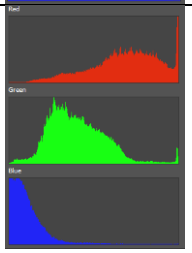
3.1.1 Pengujian *Encode* dengan Jumlah Karakter Pesan yang Sama

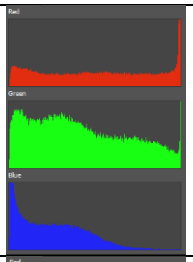
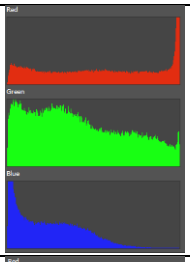
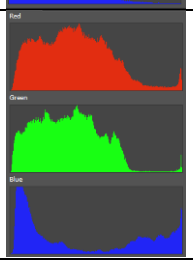
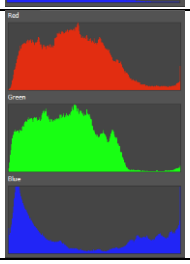
Pada pengujian ini, disediakan 5 citra dengan format jpg (*.jpg) dengan ukuran resolusi berbeda yang diberi perlakuan yang sama yaitu disisipkan pesan sebanyak 500 karakter dan diberi *password* "12345". Pengujian ini dilakukan untuk mengetahui berapa *bytes* ukuran *file* citra setelah disisipkan pesan dengan panjang karakter yang sama dan berapa lama waktu *encode* yang diperlukan.

Tabel 2 Pengujian *Encode* Pesan dengan Jumlah Karakter yang Sama

No.	**Nama *File***	**Ukuran Resolusi (*Pixel*)**	**Ukuran *Cover-image* (*Bytes*)**	**Waktu *Encode* (mili sekon)**	**Ukuran Resolusi Sesudah (*Pixel*)**	**Ukuran *Stego-image* (*Bytes*)**
1.	Pohon.jpg	400x400	43.195	2.865	400x400	39.971
2.	Harimau.jpg	640x480	296.673	4.851	640x480	76.055
3.	View.jpg	1024x768	345.194	10.040	1024x768	158.306
4.	Buah.jpg	3264x2448	2.923.659	20.874	1632x1224	428.722
5.	Villa.jpg	4000x4000	10.218.854	13.258	1000x1000	208.918

Tabel 3 Perbandingan Histogram *Cover-Image* dan *Stego-Image*

No.	**Nama *File***	***Cover-Image***	**Histogram *Cover-image***	**Histogram *Stego-image***
1.	Pohon.jpg			
2.	Harimau.jpg			
3.	View.jpg			

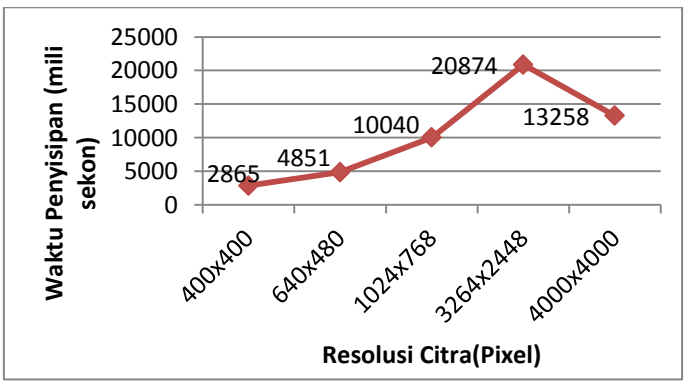
4.	Buah.jpg			
5.	Villa.jpg			

Tabel 3 menunjukkan bahwa histogram *cover-image* dan *stego-image* tidak jauh berbeda secara kasat mata. Untuk mengetahui seberapa besar perbedaan yang dihasilkan, maka dilakukan perhitungan persentase selisih rata-rata antar *cover-image* dan *stego-image* yang disajikan pada tabel 4.

Tabel 4 Persentase Selisih Rata-rata Antara Histogram *Cover-Image* dan *Stego-Image*

No.	**Nama citra**	**Rata-rata Histogram *Cover-image* (a)**	**Rata-rata Histogram *Stego-image* (b)**	**Persentase Selisih Rata-rata (%) ($c = (b - a) / b \times 100$ %)**
1	Pohon.jpg	120,08	120,32	0,20
2	Harimau.jpg	89,24	89,45	0,24
3	View.jpg	98,75	99,39	0,65
4	Buah.jpg	105,37	105,31	0,06
5	Villa.jpg	94,18	94,29	0,12
jumlah		**507,62**	**508,76**	**0,22**
Rata-rata				**0,23**

Untuk mengetahui hubungan antara waktu penyisipan dan resolusi citra dapat dilihat pada gambar 7.



Gambar 7 Hubungan Antara Waktu Penyisipan dan Resolusi Citra

Pada citra resolusi 4000×4000 piksel waktu penyisipan yang dibutuhkan lebih sedikit. Hal ini dikarenakan citra yang berukuran lebih dari 1280 (*width/height*) piksel sebelum diolah akan di-*scaling* terlebih dahulu oleh aplikasi, maka dari itu ukuran resolusi citra akan mengecil.

3.1.2 Pengujian *Encode* dengan Jumlah Karakter Pesan yang Berbeda

Dalam pengujian berikut ini, disediakan satu buah citra dengan format jpg (*jpg) dengan ukuran resolusi 640 x 480 piksel dan ukuran *file* 296.673 *Bytes*. Citra tersebut kemudian diberikan perlakuan yang berbeda yaitu disisipkan pesan dengan panjang karakter yang berbeda kemudian diberikan *password* "12345". Pengujian ini dilakukan untuk melihat seberapa besar perubahan yang terjadi pada citra jika dilihat dari jumlah karakter pesan yang disisipkan.

Tabel 5 Pengujian *Encode* Pesan dengan Panjang Karakter Berbeda

No.	**Panjang Karakter**	**Waktu *Encode* (mili sekon)**	**Ukuran Resolusi Sesudah (*Pixel*)**	**Ukuran Citra Sesudah (*Bytes*)**	**Hasil *Encode***	
					Sukses	**Gagal**
1.	1	4248	640 x 480	76.252	✓	
2.	100	4376	640 x 480	76.226	✓	
3.	500	4791	640 x 480	76.055	✓	
4.	1000	5017	640 x 480	75.719	✓	
5.	2000	5277	640 x 480	75.105	✓	

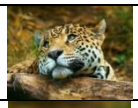
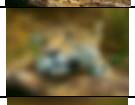
Data pada tabel 5 menunjukkan semakin banyak karakter yang disisipkan, semakin kecil ukuran *stego-image* yang dihasilkan. Hal ini disebabkan karena pada aplikasi ini menggunakan *Huffman Coding* untuk proses kompresi citra agar *stego-image* berukuran lebih kecil dari *cover-image*.

3.2 Pengujian Ketahanan Citra

Dalam pengujian ketahanan citra, disediakan 1 buah citra dengan format .jpg yang disisipkan pesan sebanyak 500 karakter dan diberi *password* "12345". Ukuran resolusi citra 640 x 480 piksel dan ukuran *file* : 76.055 *Bytes*.

Gambar 8. Citra yang Digunakan untuk Pengujian

Tabel 6 Pengujian Citra dengan Penambahan *Gaussian Blur*

No.	Radius efek (*Pixel*)	Ukuran Sesudah (KB)	*Stego-Image* Setelah Penambahan Efek	Hasil *Decode*	
				Sukses	Gagal
1.	1,0	194			✓
2.	10,0	74,6			✓
3.	50,0	63,1			✓

Tabel 7 Pengujian *Decode Stego-Image* Setelah Dirotasi

No.	Rotation Sebesar (^{o})	Ukuran Sesudah (KB)	*Stego-image* setelah di rotation	Hasil *Decode*	
				Sukses	Gagal
1.	45	370			✓
2.	90	124			✓
3.	180	124			✓

Tabel 8 Pengujian *Decode Stego-Image* Setelah Di-*scaling*

No.	*Scalling* menjadi (*Pixel*)	Ukuran Sesudah (KB)	*Stego-image* Setelah di *Scalling*	Hasil *Decode*	
				Sukses	Gagal
1.	1024x768	529			✓
2.	320x240	103			✓
3.	100x75	27,4			✓

Tabel 9 Pengujian *Stego-Image* Setelah Di-c*ropping*

No.	***Stego-image* Setelah di-*cropping***	**Ukuran Sesudah (KB)**	**Hasil *Decode***	
			Sukses	**Gagal**
1.		258		✓
2.		221		✓
3.		111		✓

Tabel 10 Pengujian Pengiriman *Stego-Image*

No.	Pengiriman Melalui	Ukuran Setelah Dikirim	Hasil *Decode*		Keterangan
			Sukses	Gagal	
1.	BBM	76.055	✓		Sukses jika penerima meminta gambar HD (fitur BBM).
2.	E-mail	76.055	✓		-
3.	Facebook	57.203		✓	Gagal karena *stego-image* mengalami kompresi ulang.

Berdasarkan hasil pengujian ketahanan *stego-image* pada tabel 6, 7, 8, 9, disimpulkan bahwa citra tidak *robust* terhadap operasi manipulasi citra. Hal ini dikarenakan operasi tersebut mempengaruhi perubahan nilai koefisien DCT sehingga tidak bisa melakukan *decode* pesan karena pesan yang berada dalam *stego-image* telah rusak. Namun dalam pengujian pengiriman pesan sesuai tabel 10, hanya media pengiriman menggunakan Facebook yang tidak dapat melakukan *decode* pesan. Hal ini dikarenakan media tersebut melakukan pengompresan ulang terhadap citra yang dikirim sehingga merubah nilai koefisien DCT.

3.3 Pengujian Nilai *Error* pada Citra

Pengujian ini dilakukan untuk mengetahui nilai *error* antara *stego-image dan cover-image* menggunakan perhitungan *Mean Square Error* (MSE) dan *Peak Signal to Noise Ratio* (PSNR). Pengujian ini dilakukan menggunakan bantuan aplikasi MATLAB dengan kode program pada gambar 9 dan 10.

```
>> origImg = imread ('C:\Users\Niko\Desktop\imagePengujian\view1024x768.jpg');
distImg= imread ('C:\Users\Niko\Desktop\imagePengujian\SteganografiF5\1421035211775.jpg');
origImg = double(origImg);
distImg = double(distImg);

[M N] = size(origImg);
error = origImg - distImg;
MSE = sum(sum(error .* error)) / (M * N);
```

Gambar 9 Kode Program Perhitungan MSE

```
>> origImg = imread ('C:\Users\Niko\Desktop\imagePengujian\view1024x768.jpg');
distImg= imread ('C:\Users\Niko\Desktop\imagePengujian\SteganografiF5\1421035211775.jpg');
origImg = double(origImg);
distImg = double(distImg);

[M N] = size(origImg);
error = origImg - distImg;
MSE = sum(sum(error .* error)) / (M * N);

if(MSE > 0)
PSNR = 10*log(255*255/MSE) / log(10);
else
PSNR = 99;
end
```

Gambar 10 Kode Program Perhitungan PSNR

Pada pengujian ini, disediakan 5 citra dengan format jpg (*.jpg) dengan ukuran resolusi berbeda yang diberi perlakuan yang sama yaitu disipkan pesan sebanyak 500 karakter dan diberi *password* "12345". Tabel 11 menunjukkan nilai *error* yang dihasilkan menggunakan perhitungan MSE dan tabel 12 menunjukkan nilai *error* yang dihasilkan menggunakan perhitungan PSNR.

Tabel 11 Pengujian Nilai *Error* Menggunakan Perhitungan MSE

No	**Nama *File***	**Ukuran Resolusi (*Pixel*)**	**Ukuran Citra (*Bytes*)**	**Waktu Penyisipan (mili sekon)**	**Ukuran Resolusi Sesudah (*Pixel*)**	**Ukuran Citra Sesudah (*Bytes*)**	**MSE**
1.	Pohon.jpg	400x400	43.195	2.865	400x400	39.971	5,939
2.	Harimau.jpg	640x480	296.673	4.851	640x480	76.055	12,1662
3.	View.jpg	1024x768	345.194	10.040	1024x768	158.306	13,7288
4.	Buah.jpg	3264x2448	2.923.659	20.874	1632x1224	428.722	-
5.	Villa.jpg	4000x4000	10.218.854	13.258	1000x1000	208.918	-

Tabel 12 Pengujian Nilai *Error* Menggunakan Perhitungan PSNR

No	Nama *File*	Ukuran Resolusi (*Pixel*)	Ukuran Citra (*Bytes*)	Waktu Penyisipan (mili sekon)	Ukuran Resolusi Sesudah (*Pixel*)	Ukuran Citra Sesudah (*Bytes*)	PSNR (dB)
1.	Pohon.jpg	400x400	43.195	2.865	400x400	39.971	41,9956
2.	Harimau.jpg	640x480	296.673	4.851	640x480	76.055	39,0912
3.	View.jpg	1024x768	345.194	10.040	1024x768	158.306	39,1841
4.	Buah.jpg	3264x2448	2.923.659	20.874	1632x1224	428.722	-
5.	Villa.jpg	4000x4000	10.218.854	13.258	1000x1000	208.918	-

Berdasarkan pengujian terhadap nilai *error* antara *stego-image* dan *cover-image* maka disimpulkan bahwa nilai MSE yang dihasilkan cukup rendah yaitu antara 5-13 dan nilai PSNR yang dihasilkan berkisar antara 39 – 41 dB yang menunjukkan bahwa *stego-image* mendekati *cover-image*. Nilai MSE pada citra Buah.jpg dan Villa.jpg tidak dapat dihitung, dikarena terjadi perubahan ukuran resolusi citra sehingga jumlah matriks tidak sama. Nilai PSNR bergantung pada nilai MSE yang dihasilkan, semakin kecil nilai *error* MSE, maka semakin besar nilai PSNR yang dihasilkan. Standar nilai PSNR yang baik yaitu diatas 30 – 40 dB.

4. KESIMPULAN

Kesimpulan yang diperoleh berdasarkan implementasi dan pengujian yang telah dilakukan, sebagai berikut:

1. Algoritme F5 dapat digunakan untuk menyisipkan pesan rahasia ke dalam citra digital berwarna (RBG) dengan format *.jpg dan dapat diterapkan untuk citra dengan resolusi 640×480 sampai dengan 1280 (*width/height*) piksel. Namun, untuk citra dengan resolusi di atas 1280 akan mengalami perubahan ukuran (*scalling*) terlebih dahulu agar dapat memenuhi syarat menjadi sebuah *cover-image*.
2. Citra yang dihasilkan (*stego-image*) secara kasat mata tidak jauh berbeda antara *cover-image* dan *stego-image* dengan angka persentase rata-rata selisih histogram *cover-image* dan *stego-image* sebesar 0,23%.
3. Waktu *encode* pesan bergantung dengan resolusi *cover-image*. Semakin besar resolusi *cover-image*, maka dibutuhkan waktu *encode* pesan lebih lama.
4. *Stego-image* tidak tahan (tidak *robust*) terhadap berbagai manipulasi citra, seperti pemberian efek *Gaussian Blur*, merotasi citra, merubah ukuran (*scalling*), dan memotong citra (*cropping*). *Stego-image* akan gagal ketika di-*decode* untuk mengambil pesan.
5. Steganografi menggunakan algoritme F5 menghasilkan *stego-image* dengan kualitas yang baik dengan nilai PSNR yang didapat untuk masing-masing citra yang diuji berkisar antara 39 dB sampai 41 dB.

5. SARAN

Penyembunyian pesan rahasia menggunakan algoritme F5 masih bisa dikembangkan untuk selanjutnya. Adapun saran untuk pengembang selanjutnya yaitu:

1. Mengembangkan aplikasi agar dapat melakukan penyisipan pesan rahasia ke dalam media selain citra dan bisa memasukkan pesan berupa *file*.
2. Mengembangkan metode penyisipan pesan ke dalam citra yang *robust* terhadap berbagai manipulasi citra dan bisa melakukan *decode* pesan.

UCAPAN TERIMA KASIH

Penulis mengucapkan puji dan syukur kepada Tuhan Yesus karena tanpa anugerah-Nya yang luar biasa, penulis tidak bisa menyelesaikan penelitian ini. Terima kasih yang sebesar-besarnya penulis ucapkan kepada orangtua yang selama ini selalu menjadi motivasi dan semangat terbesar serta selalu memberi dukungan finansial untuk penelitian ini, dosen pembimbing yang sudah meluangkan waktu untuk memberikan saran dan masukan, dosen-dosen yang selama ini turut memberi sumbangan saran, serta teman-teman yang selalu memberikan semangat yang sangat berarti bagi penulis.

DAFTAR PUSTAKA

[1] Sutoyo, T 2009, *Teori Pengolahan Citra Digital*, Andi Offset, Yogyakarta.

[2] Utomo, Tri Prasetyo, 2012, Steganografi Gambar dengan Metode Least Significant Bit untuk Proteksi Komunikasi pada Media Online, *Jurnal Teknik Informatika Fakultas Sains dan Teknologi UIN Sunan Gunung Djati Bandung*, http://jumadi.blog.ugm.ac.id/files/2012/05/trip.pdf.

[3] Nugroho, Cahyono Budi, 2011, Proses Pemampatan Citra dengan Standar Kompresi JPEG, *Jurnal Teknik Elektro Universitas Diponegoro Semarang* : http://eprints.undip.ac.id/24651/2/L2F000589_MTA.pdf.

[4] Saefullah, Asep, Himawan dan Nazori Agami, 2012, Aplikasi Steganografi untuk Menyembunyikan Teks Dalam Media Image dengan Menggunakan Metode LSB, *Seminar Nasional Teknologi Informasi dan Komunikasi Terapan 2012 (Semantik 2012)*, Semarang, 23 Juni.

[5] Iza, Dzikru Rohmatul, 2013, Steganografi pada Citra Digital Menggunakan Metode Discreate Wavelet Transform, *Jurnal Fakultas Teknik Universitas Brawijaya Malang* : http://elektro.studentjournal.ub.ac.id/index.php/teub/article/viewFile/87/55.

[6] Verma, Aayushi, 2013, Implementation of Image Steganography Using 2-Level DWT Technique, *International Journal of Computer Science and Business Informatika* : http://ijcsbi.org/index.php/ijcsbi/article/viewFile/5/1.

[7] Piarsa, I Nyoman, 2011, Steganografi pada Citra JPEG dengan Metode Sequential dan Spreading, *Jurnal Fakultas Teknik Universitas Udayana*, Vol. 2, No. 1 : http://download.portalgaruda.org/article.php?article=12805&val=922.

[8] Varghese, Sonu, Faisal K K, Vinayachandran K K, 2014, Image Security Using F5 and AES Algorithm, *Proceedings of IRF International Conference*, India, 13 April :http://iraj.in/up_proc/pdf/63-139773045996-98.pdf.

[9] Suhartono, Derwin, dkk, 2012, Aplikasi Penyembunyian Pesan pada Citra JPEG dengan Algoritma F5 dalam Perangkat Mobile Berbasis Android, *Seminar Nasional Aplikasi Teknologi Informasi 2012 (SNATI 2012)*, Yogyakarta, 15-16 Juni.

[10] Zulfikar, Dian Hafidh, 2010, Uji Ketahanan Algoritma F5 pada Stego Image Terhadap Image Distortion, *Skripsi*, Fakultas Sains dan Teknologi, Universitas Islam Negeri (UIN) Maulana Malik Ibrahim, Malang.

[11] Kulkarni, Medha, 2012, Hide and Seek in JPEG Images, *Jurnal Internasional Engineering Research and Application (IJERA)*, Vol. 2, Ed. 2, 1634-1637 : http://www.ijera.com/papers/Vol2_issue4/JJ2416341637.pdf.

[12] Octavianus, Christian, Galih Saiman, Afan, 2012, Perancangan Program Aplikasi Penyembunyian Pesan pada Citra JPEG dengan Algoritma F5 dalam Perangkat Mobile Berbasis Android, *Skripsi*, Fakultas Ilmu Komputer, BINUS University, Jakarta.

[13] Mazumder, Juned Ahmed, K. Hemachandran, 2013, Study of Image Steganography Using LSB, DFT, and DWT, *International Journal of Computers and Technology* : http://ijctonline.com/ojs/index.php/ijct/article/download/2545/pdf_275.